DD2421 Machine Learning Lab 2

Yunsen Xing

September 29, 2025

Support Vector Machine

- x $= \begin{bmatrix} x1 \\ x2 \\ ... \\ x_n \end{bmatrix}$, w $= \begin{bmatrix} w1 \\ w2 \\ ... \\ w_n \end{bmatrix}$, $X = (X_1, X_2, X_3, ..., X_n)$, $W = (w_1, w_2, w_3..., w_n)$
- decision boundary: $w_1 x_1 + w_2 x_2 + ... + w_n x_n + b = 0$
- Vector format: $W^T x + b = 0$
- The SVM algorithm attempts to find an optimal decision boundary that maximizes the distance from the nearest samples of each class.
- Suppose the shortest of these distances is d, then the margin is defined as the length of 2d.

Support Vector Machine

- The distance from the i-th sample to the hyperplane M is:
- $distance(X^i, M) = \frac{\left|w \cdot x^i + b\right|}{\|w\|}$

$$\begin{cases} \dfrac{\left|w \cdot x^i + b\right|}{\|w\|} \geq d, \quad y^i = 1 \;\rightarrow\; \text{positive}, \\ \\ \dfrac{\left|w \cdot x^i + b\right|}{\|w\|} \leq -d, \quad y^i = -1 \;\rightarrow\; \text{negative}. \end{cases}$$

$$\begin{cases} \frac{\left|w \cdot x^i + b\right|}{\|w\| \cdot d} \geq 1, \\ \\ \frac{\left|w \cdot x^i + b\right|}{\|w\| \cdot d} \leq -1, \end{cases}$$

Support Vector Machine

- *let* $w_d = \frac{w}{|w| \cdot d}$
- *let* $b_d = \frac{b}{|w| \cdot d}$
- Therefore, the constraint becomes: $\begin{cases} w_d^T x + b_d \geq 1 \\ w_d^T x + b_d \leq -1 \end{cases}$
- For the positive support vectors: $w^T x + b = 1$
- For the negative support vectors: $w^T x + b = -1$
- $\mathrm{d} = \frac{w^T x + b}{w} \rightarrow$ *for the positive support vector* : $\mathrm{d} = \frac{1}{w}$
- $\mathrm{margin} = 2\mathrm{d} = \frac{2}{||w||}$
- So minimize $\frac{1}{2}\|w\|^2$:

$$
\begin{aligned}
&= \frac{1}{2}(w_1^2 + w_2^2 + w_3^2 + \cdots + w_n^2) \\
&= \frac{1}{2} w^T w
\end{aligned}
$$

Support Vector Machine

Step:

- Minimize $\frac{1}{2}\|w\|^2$ and subject to $y^i(wx_i + b) \geq 1$
- Format training samples (x, y) appropriately.
- Use Scipy to solve for support vectors and weights.
- Construct the decision function.
- Classify new data.
- Map the data into a higher-dimensional space via a nonlinear transformation, then separate it with a linear hyperplane that maximizes the margin.

Indicator Function

- A new sample is classified using the function:

$$ind(\vec{s}) = w^T \phi(s) - b$$

- If the result is positive $\rightarrow$ *the sample is classified as positive*.
- If the result is negative $\rightarrow$ *the sample is classified as negative*.

Dual Formulation

- in the original problem, directly optimizing w and b can be complex.
- By converting it into the dual formulation, the optimization variables become the α_i
- With the kernel trick, there is no need to compute the high-dimensional mapping $\phi(x)$. Instead, it is sufficient to compute the kernel function $K(x_i, x_j)$, which enables efficient classification in a high-dimensional space.

Dual Formulation

Step:

- Minimize $\frac{1}{2}\|w\|^2$
- Constrains: $\forall i, t_i(w^T\phi(x_i) - b) - 1 \geq 0$
- Introduce Lagrange multipliers $(\alpha_i \geq 0)$
- $L(w, b, \alpha) = \frac{1}{2}\|w\|^2 - \sum \alpha[t_i(w^T\phi(x_i - b)) - 1]$
- Differentiating with respect to w:

$$\frac{\partial L}{\partial w} = w - \sum_i \alpha_i t_i \phi(x_i) = 0$$

$$w = \sum_i \alpha_i t_i \phi(x_i)$$

- Differentiating with respect to b:

$$\frac{\partial L}{\partial b} = \sum_i \alpha_i t_i = 0$$

Dual Formulation

- Substitute into the original formula:

$$L(\alpha) = \frac{1}{2}\|w\|^2 - \sum_i \alpha_i t_i \, w^T \phi(x_i) + \sum_i \alpha_i t_i b + \sum_i \alpha_i$$

$$\begin{aligned}
\|w\|^2 &= w^T w \\
&= \Big(\sum_i \alpha_i t_i \phi(x_i)\Big)^T \Big(\sum_j \alpha_j t_j \phi(x_j)\Big) \\
&= \sum_i \sum_j \alpha_i \alpha_j t_i t_j \, \phi(x_i)^T \phi(x_j)
\end{aligned}$$

$$\sum_i \alpha_i t_i = 0$$

Dual Formulation

Calculate the $L(\alpha)$:

$$
\begin{aligned}
L(\alpha) &= \tfrac{1}{2}\|w\|^2 - \sum_i \alpha_i t_i \, w^T \phi(x_i) + \sum_i \alpha_i t_i b + \sum_i \alpha_i \\
&= L(\alpha) = \tfrac{1}{2}\|w\|^2 - \|w\|^2 + \sum_i \alpha_i \\
&= -\frac{1}{2}\|w\|^2 + \sum_i \alpha_i \\
&= \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j t_i t_j K(\vec{x_i}, \vec{x_j}) \\
P_{ij} &= t_i t_j K(\vec{x_i}, \vec{x_j})
\end{aligned}
$$

Dual Formulation

- Only support vectors have non-zero α
- Most α's are zero
- Support vectors lie on the boundary and satisfy the following constraints:

$$f(x_s) = t_s \quad (t_s \in -1, +1)$$

$$f(x_s) = \sum_i \alpha_i t_i K(X_s, X_i) - b$$

$$b = \sum_i \alpha_i t_i K(X_s, X_i) - f(x_s)$$

$$b = \sum_i \alpha_i t_i K(X_s, X_i) - t_s$$

Once we optimize the dual formulation and obtain the optimal α values and b, we can construct the indicator function to classify a new sample.

Kernel Function

- $K(x_i, x_j)$ *is called kernel function*
- $K(x_i, x_j) = \phi(x_i)\phi(x_j)$
- The kernel function can be used to calculate the high-dimensional mapping(the dot product between $\phi(x_i)\phi(x_j)$), without explicitly computing $\phi(x_i)\phi(x_j)$

Indicator Function

Calculate the Indicator Function:

$$
\begin{aligned}
f(x) &= w^T\phi(x) - b \quad (w = \sum_i \alpha_i t_i \phi(x_i)) \\
f(x) &= (\sum_i \alpha_i t_i \phi(x_i))^T \phi(s) - b \\
&= \sum_i \alpha_i t_i \phi(x_i)\phi(s) - b \\
&= \sum_i \alpha_i t_i K(x_i, s) - b
\end{aligned}
$$

Slack Variables

- $\forall i, t_i(w^T\phi(x_i) - b) \geq 1 - \epsilon_i$
- if $\epsilon = 0$: Point is on the correct side, with a margin of at least 1.
- if $0 < \epsilon < 1$: Point is on the correct side, lying within the margin region.
- if $1 < \epsilon$: Point is misclassified.
- Instead of requiring that every data point is outside the margin, we will now allow for mistakes, quantified by variables ϵ. These are called slack variables. The constraints will now be:

$$f(x) = t_i(\vec{w} \cdot \phi(\vec{x_i} - b) \geq 1 - \epsilon, \forall i$$

- Soft-Margin SVM:

$$Minimize : \frac{1}{2}\|w\|^2 + C\sum_i \epsilon_i$$

- Constrains:

$$0 \leq \alpha \leq C \ and \ \sum_i \alpha_i t_i = 0$$

Assignment 1

- Changing the values of x and y in *numpy.random.randn*$(10, 2)$ moves the cluster. When the clusters of class A and B overlap, the optimizer using a linear kernel function is unable to find a solution.

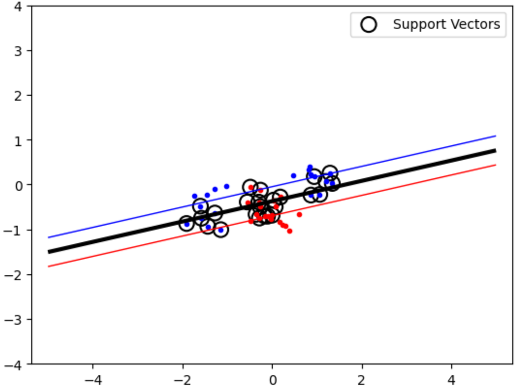

Figure: overlapping

Assignment 2 - kernel function - Polynomial kernels

- This kernel allows for curved decision boundaries. The exponent p (a positive integer) controls the degree of the polynomials.

$$K(\vec{x}, \vec{y}) = (\vec{x}^T \cdot \vec{y} + 1)^p$$

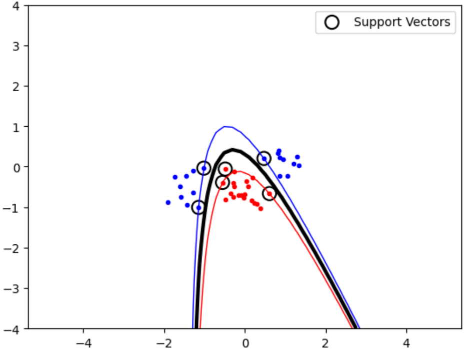

Figure: p = 2

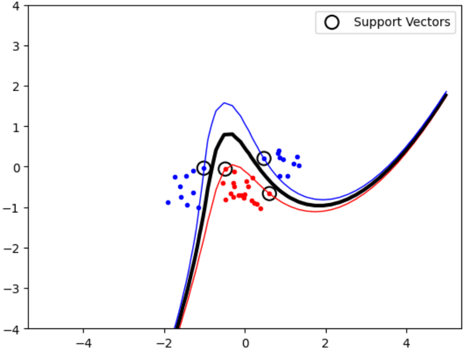

Figure: p = 3

Assignment 2 - Radial Basis Function & Assignment 3

- This kernel uses the explicit Euclidean distance between the two data points, and often results in very good boundaries. The parameter σ is used to control the smoothness of the boundary.
- decreasing sigma may lead to overfitting
- increasing sigma leads to smoother boundary considerations and may lead to better generic results.

$$\mathcal{K}(\vec{x}, \vec{y}) = e^{-\frac{\|\vec{x}-\vec{y}\|^2}{2\sigma^2}}$$

Assignment 2 & Assignment 3

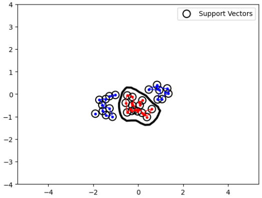

Figure: $\sigma = 0.2$

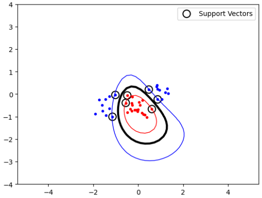

Figure: $\sigma = 1$

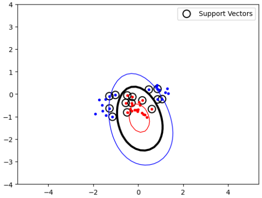

Figure: $\sigma = 2$

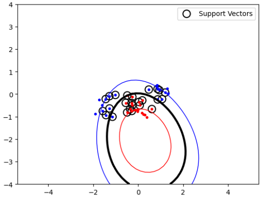

Figure: $\sigma = 3$

Assignment 4

- The parameter C controls the balance between minimizing slack and maximizing the margin.
- Large C $\rightarrow$ strict separation and narrow margin.
- Small C $\rightarrow$ more slack and wider margin.

Assignment 4

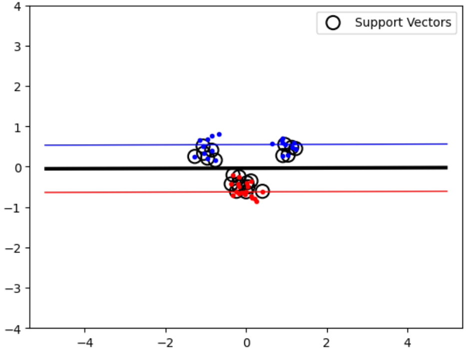

Figure: C = 0.2

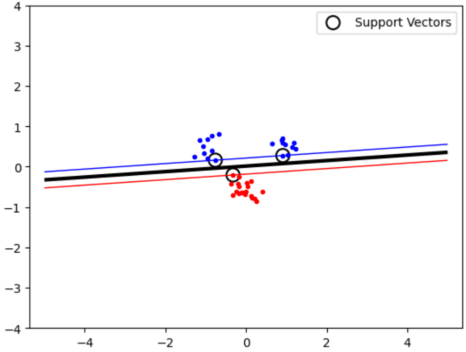

Figure: C = 1000

Assignment 5

- If the data is noisy, it is better to use more slack so that the classifier does not overfit.
- If the data is not linearly separable because of its structure, it is better to use a more complex kernel to separate the data